IN THE SPECIFICATION

Please amend the paragraph beginning on page 1, line 7, as follows:

The present invention relates generally to the field of software systems that manage the life-cycle of software applications, e.g., installation, configuration, resource management, security management, execution, and de-installation, and, more particularly, to methods and apparatus for processing of web applications that are written in the form of web pages which can be downloaded through the Internet using web communication protocols, installed in local computers, and executed utilizing web components, such as a web browser and JavaScript™ interpreter, in the local computers.

Please amend the consecutive paragraphs beginning on page 1, line 21, as follows:

Using the Web, users can access remote information by receiving web pages through the Hypertext Transfer Protocol (HTTP). The information in a web page is described using the Hypertext Markup Language (HTML) and eXtensive Markup Language (XML), and is displayed by software called web browser. Web pages of earlier design are considered static because they do not include any logic that can dynamically change their appearances or provide computations based on user input. Subsequently, the Java™ (Sun Microsystems) programming language was incorporated in web pages in the form of applets. An applet is a small Java™ program that can be sent along with a web page to a user. Java™ applets can perform interactive animation, immediate calculations, or other simple tasks without having to send a user request back to the server, thereby providing the dynamic logic in web pages.

Java™ is an object-oriented programming language which can be used for creating stand-alone applications. Writing Java™ programs typically requires different and more extensive skills and training than composing web pages. The learning curve for writing Java™ programs is typically longer than that for writing web pages. Not all web page authors therefore are expert Java™ programmers.

Recently, to make it easier to embed logic in web pages, an easy-to-write script language called JavaScript™ (Sun Microsystems) has been supported by popular web browsers to be incorporated into web pages. JavaScript™, capable of embedding logic for computation based on user input, brings dynamic and powerful capabilities to web pages. JavaScript™, unlike Java™

which is a full-fledged programming language, has a simpler syntax and is much easier to learn. Because of this easy-to-write feature, JavaScript™ has currently become a popular way to embed logic in web pages by many web page authors.

Although JavaScript™ brings easy-to-write logic to web pages, it is limited to browser functions and works with HTML elements only. It can only be used to create simple applications under the contexts of the browser, such as changing the web page's visual presentation dynamically and computing user input quickly without sending a user request back to the server (for such computation). Thus, web pages with JavaScript™ logic cannot be used to create stand-alone applications that require access to a full range of resources on the user's computer such as the file system management and the display area beyond the browser's window. In general, web pages cannot be processed in non-browser contexts.

At the present, stand-alone applications are typically written in traditional programming languages (also called 3GL for 3rd Generation Languages) such as C, C++, and Java™, or Fourth Generation Languages (4GL) such as Visual Basic™. Through these languages, stand-alone applications interact directly with operating systems through operating system APIs (application programming interfaces) or indirectly with library functions which may in turn call these operating system APIs. The capability of accessing the operating system APIs gives an application the control of computing resources in a computer.

Please amend the paragraph beginning on page 5, line 27, as follows:

In a virtual machine environment, such as the Java™ Virtual Machine, the security context of an application (such a Java™ program) is defined by the virtual machine. A misbehaving application thus can only create external damage allowable by the virtual machine. However, there can be many different types of applications running on the same virtual machine and while each one of them may have a different security need, they are forced to run under the same security context (that defined by the virtual machine).

Please amend the consecutive paragraphs beginning on page 7, line 24, as follows:

In accordance with the aforementioned needs, the present invention is directed to a system in which applications are written as web pages that have access to the full range of operating system

resources, including those not accessible through the web browser. The applications described in the present invention are called web applications. In a preferred embodiment of the present invention, three types of languages used for constructing web pages are used for building web applications. They are: (1) a visual presentation language; (2) a data modeling language; and (3) a scripting language for embedding logic. Those skilled in the art will appreciate that currently the three most commonly used languages in web pages are HTML for visual presentation, XML for data modeling, and JavaScript™ for scripting.

According to the present invention, a software system is provided to allow a computer to install and process web applications. This system preferably comprises a web application manager, an operating system interface module, a scripting language interpreter, and optionally a web browser and/or a data modeling language processor. The web application manager manages the life-cycle for applications, which may include the installation, execution, de-installation of these applications, as well as the security control and web caching for these applications. The script language interpreter (such as the JavaScript™ interpreter) parses and interprets the scripting language embedded in the web pages. The operating system interface module is used to convert the scripting language calls that request access to system resources into appropriate native operating system APIs. The web browser can be used to display the content of web applications and transfer data based on the data transfer protocol deployed by the browser (such as HTTP). The data modeling language processor (such as the XML parser) decodes the contents in the web applications that are written in the data modeling language (such as XML).

Please amend the paragraph beginning on page 18, line 23, as follows:

FIG. 9 depicts an example 901 of an install document of an application used in accordance with the present invention to properly install this application. As depicted by FIG. 9, the install document of an application may include, but is not limited to, general information, various required components, dependency information, registry information, short cut information, storage quota, and security settings of this application.

Please amend the consecutive paragraphs beginning on page 21, line 10, as follows:

FIG. 14 depicts the security context of virtual machine (VM) based applications. As depicted in FIG. 14, direct calling of operating system 1406 APIs from the applications is eliminated. Instead, applications 1401 request system resources through calls made to the virtual machine 1402. It is the virtual machine 1402 that makes the operating systems' APIs access operating system services such as the file system 1403, network services 1404, process services 1405, and other operating system services on behalf of the applications.

In a virtual machine environment, such as the Java™ Virtual Machine, the security context of an application (such as a Java™ program) is therefore defined by the virtual machine. A misbehaving application thus can only create external damage allowable by the virtual machine. However, there can be many different types of applications running on the same virtual machine and while each one of them may have a different security need, they are forced to run under the same security context (that defined by the virtual machine).

Please amend the paragraph beginning on page 23, line 21, as follows:

FIG. 18 depicts an architecture of a web caching system of the present invention. As depicted in FIG. 18, web objects (1801) are retrieved via the Internet by the web application manager (1804). For each web application (1802 or 1803), the web application manager creates a separate web cache (1807 or 1808). In a preferred embodiment of the present invention, the web cache for an application contains two pools, one for HTTP objects and the other one for non-HTTP objects. In this preferred embodiment, the web application manager runs a separate copy of the web browser software (1805 or 1806) for each web application, and uses the web browser's web caching system to cache HTTP objects for each application. Those skilled in the art can appreciate that browser software can be incorporated by the present invention using various methods, including for example linking and invoking APIs calls they provide or incorporating their source code for compilation. Alternatively, the web application manager can be developed with the capabilities to create and manage a cache for HTTP objects for each application without the incorporation of the browser's caching system.